In what precision(s) should one precondition?

Theo Mary
Sorbonne Université, CNRS, LIP6

Preconditioning 2026
Edinburgh, 27 May 2026

Why precision matters for preconditioning

Standard model of floating-point arithmetic:

For any x such that $|x| \in [f_{\min}, f_{\max}]$,
$$\mathrm{fl}(x) = x(1+\delta), \quad |\delta| \le u$$

		Range $f_{\max}/f_{\min}$	Unit roundoff u
fp64	e11m52	$2^{2046} \approx 10^{616}$	$2^{-53} \approx 1 \times 10^{-16}$
fp32	e8m23	$2^{254} \approx 10^{76}$	$2^{-24} \approx 6 \times 10^{-8}$

- Preconditioners are approximate: avoid oversolving with needlessly high precision

Why precision matters for preconditioning

Standard model of floating-point arithmetic:

For any x such that $|x| \in [f_{\min}, f_{\max}]$,
$$\mathrm{fl}(x) = x(1+\delta), \quad |\delta| \le u$$

		Range $f_{\max}/f_{\min}$	Unit roundoff u
fp64	e11m52	$2^{2046} \approx 10^{616}$	$2^{-53} \approx 1 \times 10^{-16}$
fp32	e8m23	$2^{254} \approx 10^{76}$	$2^{-24} \approx 6 \times 10^{-8}$
fp16	e5m10	$2^{30} \approx 10^{9}$	$2^{-11} \approx 5 \times 10^{-4}$
bfloat16	e8m7	$2^{254} \approx 10^{76}$	$2^{-8} \approx 4 \times 10^{-3}$
fp8	e4m3	$2^{15} \approx 3 \times 10^{4}$	$2^{-4} \approx 6 \times 10^{-2}$
fp8	e5m2	$2^{30} \approx 10^{9}$	$2^{-3} \approx 1 \times 10^{-1}$
fp6	e2m3	$2^{3} \approx 8$	$2^{-4} \approx 6 \times 10^{-2}$
fp6	e3m2	$2^{7} \approx 128$	$2^{-3} \approx 0.125$
fp4	e2m1	$2^{3} \approx 8$	$2^{-2} \approx 0.25$

- Preconditioners are approximate: avoid oversolving with needlessly high precision
- The emergence of (very) low precisions means we cannot ignore rounding errors any more

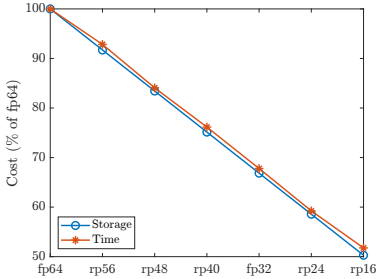
Why precision matters for preconditioning

Standard model of floating-point arithmetic:

For any x such that $|x| \in [f_{\min}, f_{\max}]$,
$\mathrm{fl}(x) = x(1+\delta), \quad |\delta| \le u$

		Range $f_{\max}/f_{\min}$	Unit roundoff u
fp64	e11m52	$2^{2046} \approx 10^{616}$	$2^{-53} \approx 1 \times 10^{-16}$
rp56	e11m44	$2^{2046} \approx 10^{616}$	$2^{-45} \approx 3 \times 10^{-14}$
rp48	e11m36	$2^{2046} \approx 10^{616}$	$2^{-37} \approx 7 \times 10^{-12}$
rp40	e11m28	$2^{2046} \approx 10^{616}$	$2^{-29} \approx 2 \times 10^{-9}$
fp32	e8m23	$2^{254} \approx 10^{76}$	$2^{-24} \approx 6 \times 10^{-8}$
rp24	e8m15	$2^{254} \approx 10^{76}$	$2^{-16} \approx 2 \times 10^{-5}$
fp16	e5m10	$2^{30} \approx 10^{9}$	$2^{-11} \approx 5 \times 10^{-4}$
bfloat16	e8m7	$2^{254} \approx 10^{76}$	$2^{-8} \approx 4 \times 10^{-3}$
rp16	e8m7	$2^{254} \approx 10^{76}$	$2^{-8} \approx 4 \times 10^{-3}$
fp8	e4m3	$2^{15} \approx 3 \times 10^{4}$	$2^{-4} \approx 6 \times 10^{-2}$
fp8	e5m2	$2^{30} \approx 10^{9}$	$2^{-3} \approx 1 \times 10^{-1}$
fp6	e2m3	$2^{3} \approx 8$	$2^{-4} \approx 6 \times 10^{-2}$
fp6	e3m2	$2^{7} \approx 128$	$2^{-3} \approx 0.125$
fp4	e2m1	$2^{3} \approx 8$	$2^{-2} \approx 0.25$

- Compressed data formats and efficient on-the-fly decompression (memory accessors) provide a continuum of precisions with performance $\propto$ storage for memory-bound operations $\Rightarrow$ **balance precision and preconditioner's errors**



SpMV performance with memory accessor
Long_Coup_dt0 matrix

Part I

Uniform precision preconditioning

Preconditioned GMRES

$$M^{-1}Ax = M^{-1}b$$

Left GMRES

$r_0 = b - Ax_0$
$s_0 = M^{-1}r_0$
$\beta = \|s_0\|,\ v_1 = s_0/\beta,\ k = 1$
repeat
  $z_k = Av_k$
  $w_k = M^{-1}z_k$
  for $i = 1, \ldots, k$ **do**
    $h_{i,k} = v_i^T w_k$
    $w_k = w_k - h_{i,k}v_i$
  end for
  $h_{k+1,k} = \|w_k\|, v_{k+1} = w_k/h_{k+1,k}$
  $V_k = [v_1, \ldots, v_k]$
  $H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$
  $y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$
  $k = k + 1$
until $\|\beta e_1 - H_k y_k\| \le \tau$

$x_k = x_0 + V_k y_k$

$$AM^{-1}y = b,\ x = M^{-1}y$$

Right GMRES

$r_0 = b - Ax_0$

$\beta = \|r_0\|,\ v_1 = r_0/\beta,\ k = 1$
repeat
  $z_k = M^{-1}v_k$
  $w_k = Az_k$
  for $i = 1, \ldots, k$ **do**
    $h_{i,k} = v_i^T w_k$
    $w_k = w_k - h_{i,k}v_i$
  end for
  $h_{k+1,k} = \|w_k\|, v_{k+1} = w_k/h_{k+1,k}$
  $V_k = [v_1, \ldots, v_k]$
  $H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$
  $y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$
  $k = k + 1$
until $\|\beta e_1 - H_k y_k\| \le \tau$
$d_k = V_k y_k$
$x_k = x_0 + M^{-1}d_k$

$$M^{-1}Ax = M^{-1}b$$

Left GMRES

$r_0 = b - Ax_0$
$s_0 = M^{-1}r_0$
$\beta = \|s_0\|,\ v_1 = s_0/\beta,\ k = 1$
repeat
  $z_k = Av_k$
  $w_k = M^{-1}z_k$
  for $i = 1, \ldots, k$ **do**
    $h_{i,k} = v_i^T w_k$
    $w_k = w_k - h_{i,k}v_i$
  end for
  $h_{k+1,k} = \|w_k\|, v_{k+1} = w_k/h_{k+1,k}$
  $V_k = [v_1, \ldots, v_k]$
  $H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$
  $y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$
  $k = k + 1$
until $\|\beta e_1 - H_k y_k\| \le \tau$

$x_k = x_0 + V_k y_k$

$$AM^{-1}y = b,\ x = M^{-1}y$$

Flexible GMRES

$r_0 = b - Ax_0$

$\beta = \|r_0\|,\ v_1 = r_0/\beta,\ k = 1$
repeat
  $z_k = M^{-1}v_k$
  $w_k = Az_k$
  for $i = 1, \ldots, k$ **do**
    $h_{i,k} = v_i^T w_k$
    $w_k = w_k - h_{i,k}v_i$
  end for
  $h_{k+1,k} = \|w_k\|, v_{k+1} = w_k/h_{k+1,k}$
  $V_k = [v_1, \ldots, v_k],\ Z_k = [z_1, \ldots, z_k]$
  $H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$
  $y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$
  $k = k + 1$
until $\|\beta e_1 - H_k y_k\| \le \tau$

$x_k = x_0 + Z_k d_k$

Setup vs Apply

- Using low precision for setting up the preconditioner is common practice (though determining what level of precision is needed may not be obvious and is preconditioner-dependent; more on this in Part II)
- It is much less clear what precision to use for applying the preconditioner

Setup vs Apply

- Using low precision for setting up the preconditioner is common practice (though determining what level of precision is needed may not be obvious and is preconditioner-dependent; more on this in Part II)
- It is much less clear what precision to use for applying the preconditioner
- Some studies apply it in higher precision to improve accuracy. They are dedicated to left-preconditioned GMRES

E. Carson and N. J. Higham. A New Analysis of Iterative Refinement and Its Application to Accurate Solution of Ill-Conditioned Sparse Linear Systems. SISC 2017.

E. Carson and N. J. Higham. Accelerating the solution of linear systems by iterative refinement in three precisions. SISC 2018.

E. Carson, N. J. Higham, and S. Pranesh. Three-Precision GMRES-Based Iterative Refinement for Least Squares Problems. SISC 2020.

P. R. Amestoy, A. Buttari, N. J. Higham, J.-Y. L'Excellent, T. M., and B. Vieublé. Five-Precision GMRES-Based Iterative Refinement. SIMAX 2024.

P. R. Amestoy, A. Buttari, N. J. Higham, J.-Y. L'Excellent, T. M., and B. Vieublé. Combining sparse approximate factorizations with mixed-precision iterative refinement. TOMS 2023.

E. Carson and N. Khan. Mixed Precision Iterative Refinement with Sparse Approximate Inverse Preconditioning. SISC 2023.

Setup vs Apply

- Using low precision for setting up the preconditioner is common practice (though determining what level of precision is needed may not be obvious and is preconditioner-dependent; more on this in Part II)
- It is much less clear what precision to use for applying the preconditioner
- Some studies apply it in higher precision to improve accuracy. They are dedicated to left-preconditioned GMRES
- Other studies apply it in lower precision to improve performance. They are dedicated to flexible GMRES

M. Arioli and I. S. Duff. Using FGMRES to obtain backward stability in mixed precision. ETNA 2008.
J. D. Hogg and J. A. Scott. A fast and robust mixed-precision solver for the solution of sparse symmetric linear systems. TOMS 2010.
E. Carson and I. Daužickaitė. The stability of split-preconditioned FGMRES in four precisions. ETNA 2024.
E. Carson and I. Daužickaitė. Mixed precision sketching for least-squares problems and its application in GMRES-based iterative refinement. SIMAX 2025.

$\Rightarrow$ Need better understanding of the effect of Apply's precision and of the preconditioning style (left, right, flexible)

Mixed precision preconditioned GMRES

Left GMRES

1:	$r_0 = b - Ax_0$	u_a
2:	$s_0 = M^{-1} r_0$	u_m
3:	$\beta = \|s_0\|,\ v_1 = s_0/\beta,\ k = 1$	u_g
4:	**repeat**	
5:	$z_k = Av_k$	u_a
6:	$w_k = M^{-1} z_k$	u_m
7:	**for** $i = 1, \ldots, k$ **do**	
8:	$h_{i,k} = v_i^T w_k$	u_g
9:	$w_k = w_k - h_{i,k} v_i$	u_g
10:	**end for**	
11:	$h_{k+1,k} = \|w_k\|, v_{k+1} = w_k / h_{k+1,k}$	u_g
12:	$V_k = [v_1, \ldots, v_k]$	
13:	$H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$	
14:	$y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$	u_g
15:	$k = k + 1$	
16:	**until** $\|\beta e_1 - H_k y_k\| \le \tau$	
17:		
18:	$x_k = x_0 + V_k y_k$	u_g

Right GMRES

1:	$r_0 = b - Ax_0$	u_a
2:		
3:	$\beta = \|r_0\|,\ v_1 = r_0/\beta,\ k = 1$	u_g
4:	**repeat**	
5:	$z_k = M^{-1} v_k$	u_m
6:	$w_k = Az_k$	u_a
7:	**for** $i = 1, \ldots, k$ **do**	
8:	$h_{i,k} = v_i^T w_k$	u_g
9:	$w_k = w_k - h_{i,k} v_i$	u_g
10:	**end for**	
11:	$h_{k+1,k} = \|w_k\|, v_{k+1} = w_k / h_{k+1,k}$	u_g
12:	$V_k = [v_1, \ldots, v_k]$	
13:	$H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$	
14:	$y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$	u_g
15:	$k = k + 1$	
16:	**until** $\|\beta e_1 - H_k y_k\| \le \tau$	
17:	$d_k = V_k y_k$	u_g
18:	$x_k = x_0 + M^{-1} d_k$	u_m

Mixed precision preconditioned GMRES

Left GMRES

1:	$r_0 = b - Ax_0$	u_a
2:	$s_0 = M^{-1} r_0$	u_m
3:	$\beta = \|s_0\|,\ v_1 = s_0/\beta,\ k = 1$	u_g
4:	**repeat**	
5:	$z_k = Av_k$	u_a
6:	$w_k = M^{-1} z_k$	u_m
7:	**for** $i = 1, \ldots, k$ **do**	
8:	$h_{i,k} = v_i^T w_k$	u_g
9:	$w_k = w_k - h_{i,k} v_i$	u_g
10:	**end for**	
11:	$h_{k+1,k} = \|w_k\|, v_{k+1} = w_k / h_{k+1,k}$	u_g
12:	$V_k = [v_1, \ldots, v_k]$	
13:	$H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$	
14:	$y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$	u_g
15:	$k = k + 1$	
16:	**until** $\|\beta e_1 - H_k y_k\| \le \tau$	
17:		
18:	$x_k = x_0 + V_k y_k$	u_g

Flexible GMRES

1:	$r_0 = b - Ax_0$	u_a
2:		
3:	$\beta = \|r_0\|,\ v_1 = r_0/\beta,\ k = 1$	u_g
4:	**repeat**	
5:	$z_k = M^{-1} v_k$	u_m
6:	$w_k = Az_k$	u_a
7:	**for** $i = 1, \ldots, k$ **do**	
8:	$h_{i,k} = v_i^T w_k$	u_g
9:	$w_k = w_k - h_{i,k} v_i$	u_g
10:	**end for**	
11:	$h_{k+1,k} = \|w_k\|, v_{k+1} = w_k / h_{k+1,k}$	u_g
12:	$V_k = [v_1, \ldots, v_k],\ Z_k = [z_1, \ldots, z_k]$	
13:	$H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$	
14:	$y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$	u_g
15:	$k = k + 1$	
16:	**until** $\|\beta e_1 - H_k y_k\| \le \tau$	
17:		
18:	$x_k = x_0 + Z_k d_k$	u_m

Attainable accuracy of mixed precision preconditioned GMRES

Disclaimer: the following only says something about the attainable accuracy. The effect on convergence rate is a much harder problem.

Attainable forward error bounds (Buttari, Liu, M., Vieublé, 2026)

- Left: $\kappa(M^{-1}A)\boldsymbol{u_g} + \max(\rho, \kappa(M^{-1}A))\boldsymbol{u_m} + \kappa(A)\boldsymbol{u_a}$
  $(\rho \leq \kappa(M^{-1}A)\kappa(M)\|Av_j\|/\|A\|)$
- Right: $\kappa(AM^{-1})\kappa(M)\boldsymbol{u_g} + \kappa(M)\boldsymbol{u_m} + \kappa(A)\boldsymbol{u_a}$
- Flexible: $\kappa(AM^{-1})\kappa(M)\boldsymbol{u_g} + \kappa(A)\boldsymbol{u_a}$

Key observations

- Different operations and different preconditioning styles have different effects on the attainable accuracy
- Apply's precision $\boldsymbol{u_m}$ affects Left/Right's accuracy, but not Flexible's
  - Intuitive explanation: rounding errors in Apply are different at each iteration, making the preconditioner nonconstant

Flexible vs non-flexible+accessor (ongoing work w/ Amestoy, El Habti, Jego, L'Excellent)

Flexible GMRES

👎 High memory overhead (Krylov basis and its preconditioned counterpart)

👍 Preconditioner can be applied in low precision

Classical GMRES

👍 Half the memory overhead of Flexible GMRES (Krylov basis only)

👎 Preconditioner has to be applied in high precision $\Rightarrow$ can use **memory accessors**!

Flexible vs non-flexible+accessor (ongoing work w/ Amestoy, El Habti, Jego, L'Excellent)

Flexible GMRES

👎 High memory overhead (Krylov basis and its preconditioned counterpart)

👍 Preconditioner can be applied in low precision

Classical GMRES

👍 Half the memory overhead of Flexible GMRES (Krylov basis only)

👎 Preconditioner has to be applied in high precision $\Rightarrow$ can use **memory accessors**!

Illustrative example on matrix Tangent_S (MUMPS+PETSc)

- Different **time–memory tradeoffs** can be achieved by turning various knobs: restart size, block low-rank compression threshold

(time in s., mem. in GB)

	Time-optimal			Memory-optimal		
Method	time	mem.	its.	time	mem.	its.
Flexible GMRES	**60**	**26**	5	248	**16**	178
Classical GMRES	64	34	3	**153**	**16**	62

- Flexible is better to minimize time, due to memory accessor overhead
- Classical is better to minimize memory: can use twice as large restart size, which can sufficiently reduce iteration count to offset memory accessor overhead

Short-term recurrence solvers

The previous discussion is specific to GMRES. For short-term recurrence solvers, the preconditioner can often be applied in low precision

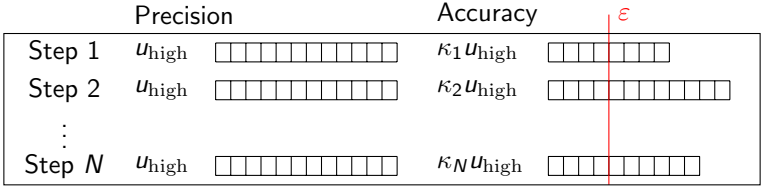
- PCG
  T. Bake, E. Carson, Y. Ma. Forward and backward error bounds for a mixed precision preconditioned conjugate gradient algorithm. arXiv:2510.11379
- BiCGSTAB
  A. Anciaux-Sedrakian, H. Dorfsman, T. Guignon, F. Jézéquel, T. M. Mixed Precision Strategies for Solving Sparse Linear Systems with BiCGStab. hal-05326901
- LSQR
  J. Scott and M. Tůma. A computational study of low precision incomplete Cholesky factorization preconditioners for sparse linear least-squares problems. arXiv:2504.07580

Part II

Mixed precision preconditioning

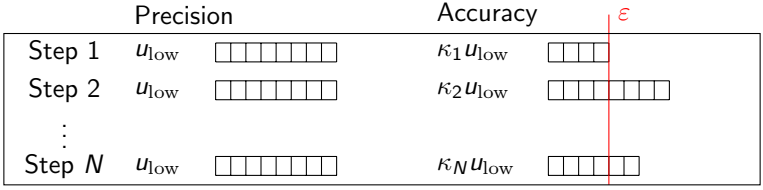
Adaptive precision (AP) algorithms

- Precision $\neq$ accuracy
  - Precision is the error of elementary operations $\{+, -, \times, \div\}$
  - Accuracy is the error of a sequence of operations
- Let $\varepsilon > 0$ be the target accuracy and let $u_1 < \ldots < u_p$ be the available precisions.
- Consider an abstract computation partitioned into N steps, and let κ_i be the precision/accuracy amplification factor of step i
- What precision should we assign to each step?



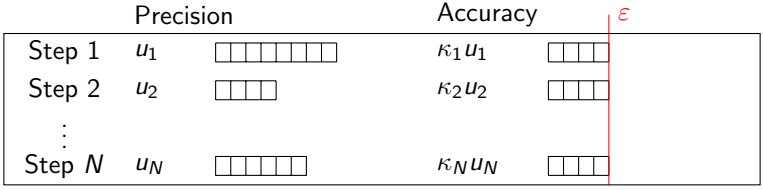
Adaptive precision (AP) algorithms

- Precision $\neq$ accuracy
  - Precision is the error of elementary operations $\{+, -, \times, \div\}$
  - Accuracy is the error of a sequence of operations
- Let $\varepsilon > 0$ be the target accuracy and let $u_1 < \ldots < u_p$ be the available precisions.
- Consider an abstract computation partitioned into N steps, and let κ_i be the precision/accuracy amplification factor of step i
- What precision should we assign to each step?
- Uniform precision: use $u_{\text{low}} \leq \varepsilon \Rightarrow$ creates imbalance, some steps are needlessly precise



Adaptive precision (AP) algorithms

- Precision $\neq$ accuracy
  - Precision is the error of elementary operations $\{+, -, \times, \div\}$
  - Accuracy is the error of a sequence of operations
- Let $\varepsilon > 0$ be the target accuracy and let $u_1 < \ldots < u_p$ be the available precisions.
- Consider an abstract computation partitioned into N steps, and let κ_i be the precision/accuracy amplification factor of step i
- What precision should we assign to each step?
- Uniform precision: use $u_{\text{low}} \leq \varepsilon \Rightarrow$ creates imbalance, some steps are needlessly precise
- Mixed precision: use u_i such that $\kappa_i u_i \approx \varepsilon$: balances errors at different steps



AP domain decomposition preconditioning (Caruso, Jolivet, M., Nataf, Tournier, 2026)

- Given an SPD system, consider the additive Schwarz preconditioner

$$M^{-1} = \sum_{i=0}^{N} R_i^T A_i^{-1} R_i,$$

where $A_1, \ldots, A_N$ are N (overlapping) subdomains and A_0 is the coarse space

- **Analysis:** the adaptive precision preconditioner

$$\widehat{M}^{-1} = \sum_{i=0}^{N} R_i^T (A_i + E_i)^{-1} R_i, \quad \|E_i\| \leq u_i \|A_i\|$$

satisfies

$$\kappa(\widehat{M}^{-1}A) \leq \frac{1+\varepsilon}{1-\varepsilon} \kappa(M^{-1}A), \quad \varepsilon = \max_i u_i \kappa(A_i)$$

AP domain decomposition preconditioning (Caruso, Jolivet, M., Nataf, Tournier, 2026)

- Given an SPD system, consider the additive Schwarz preconditioner

$$M^{-1} = \sum_{i=0}^{N} R_i^T A_i^{-1} R_i,$$

where $A_1, \ldots, A_N$ are N (overlapping) subdomains and A_0 is the coarse space

- **Analysis:** the adaptive precision preconditioner

$$\widehat{M}^{-1} = \sum_{i=0}^{N} R_i^T (A_i + E_i)^{-1} R_i, \quad \|E_i\| \leq u_i \|A_i\|$$

satisfies

$$\kappa(\widehat{M}^{-1} A) \leq \frac{1+\varepsilon}{1-\varepsilon} \kappa(M^{-1} A), \quad \varepsilon = \max_i u_i \kappa(A_i)$$

$\Rightarrow$ **Algorithm:** set $u_i \propto 1/\kappa(A_i)$

AP domain decomposition preconditioning (Caruso, Jolivet, M., Nataf, Tournier, 2026)

- Given an SPD system, consider the additive Schwarz preconditioner

$$M^{-1} = \sum_{i=0}^{N} R_i^T A_i^{-1} R_i,$$

where $A_1, \ldots, A_N$ are N (overlapping) subdomains and A_0 is the coarse space

- **Analysis:** the adaptive precision preconditioner

$$\widehat{M}^{-1} = \sum_{i=0}^{N} R_i^T (A_i + E_i)^{-1} R_i, \quad \|E_i\| \leq u_i \|A_i\|$$

satisfies

$$\kappa(\widehat{M}^{-1} A) \leq \frac{1+\varepsilon}{1-\varepsilon} \kappa(M^{-1} A), \quad \varepsilon = \max_i u_i \kappa(A_i)$$

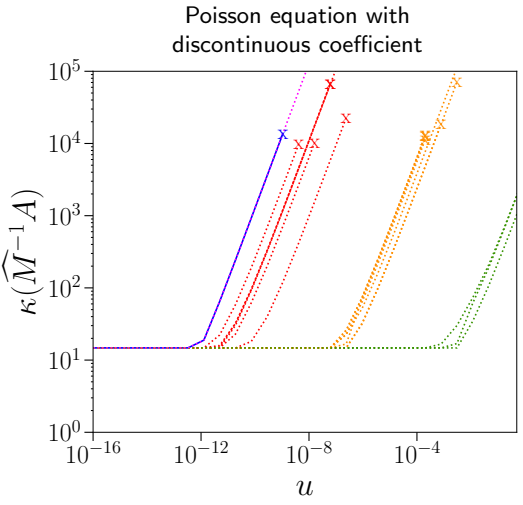
$\Rightarrow$ **Algorithm:** set $u_i \propto 1/\kappa(A_i)$

- Our analysis also covers the GenEO method to build the coarse space. It involves solving local GEVPs whose precision should also be set $\propto 1/\kappa(A_i)$

AP domain decomposition preconditioning (Caruso, Jolivet, M., Nataf, Tournier, 2026)

Uniform precision: $u_i = u$

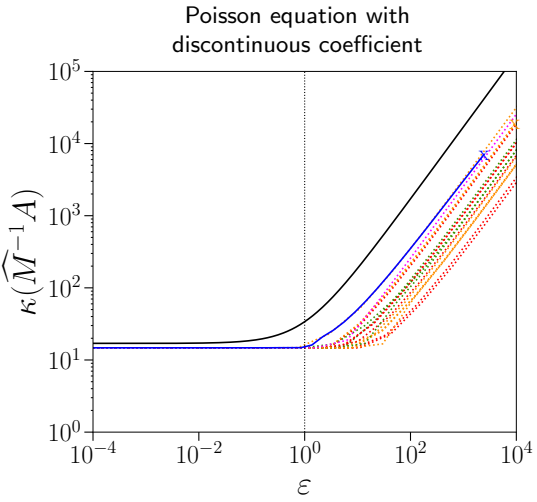
- All subdo.
- One subdo. $\kappa(A_i) \in [10^3, 10^4]$
- One subdo. $\kappa(A_i) \in [10^7, 10^9]$
- One subdo. $\kappa(A_i) \in [10^{11}, 10^{13}]$
- Coarse matrix $\kappa(A_0) = 5.0 \times 10^{12}$



AP domain decomposition preconditioning (Caruso, Jolivet, M., Nataf, Tournier, 2026)

Adaptive precision: $u_i = \varepsilon/\kappa(A_i)$

- All subdo.
- One subdo. $\kappa(A_i) \in [10^3, 10^4]$
- One subdo. $\kappa(A_i) \in [10^7, 10^9]$
- One subdo. $\kappa(A_i) \in [10^{11}, 10^{13}]$
- Coarse matrix $\kappa(A_0) = 5.0\times10^{12}$
- Theoretical bound



AP incomplete factorization (ongoing work w/ Lister and Scott)

- Let $A = LU$; ℓ_{ij} is computed as

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} u_{kj} \right) / u_{jj}$$

- **Analysis:** let $\widehat{\ell}_{ij} = \ell_{ij}(1 + \delta_{ij})$; we have

$$\ell_{ij}(1 + \delta_{ij}) = \left(a_{ij} - \sum_{k=1}^{j-1} \widehat{\ell}_{ik} \widehat{u}_{kj} \right) / \widehat{u}_{jj}$$

$$\Leftrightarrow a_{ij} = \sum_{k=1}^{j} \widehat{\ell}_{ik} \widehat{u}_{kj} + \ell_{ij} \widehat{u}_{jj} \delta_{ij}$$

AP incomplete factorization (ongoing work w/ Lister and Scott)

- Let $A = LU$; ℓ_{ij} is computed as

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} u_{kj} \right) / u_{jj}$$

- **Analysis:** let $\widehat{\ell}_{ij} = \ell_{ij}(1 + \delta_{ij})$; we have

$$\ell_{ij}(1 + \delta_{ij}) = \left(a_{ij} - \sum_{k=1}^{j-1} \widehat{\ell}_{ik} \widehat{u}_{kj} \right) / \widehat{u}_{jj}$$

$$\Leftrightarrow a_{ij} = \sum_{k=1}^{j} \widehat{\ell}_{ik} \widehat{u}_{kj} + \ell_{ij} \widehat{u}_{jj} \delta_{ij}$$

$\Rightarrow$ **Algorithm:**

$$\begin{cases} \text{if } |\ell_{ij} \widehat{u}_{jj}| \leq \varepsilon \|A\| & \text{drop } \ell_{ij} \ (\widehat{\ell}_{ij} = 0, \ \delta_{ij} = -1) \\ \text{else} & \text{store } \ell_{ij} \text{ in precision } \leq \frac{\varepsilon \|A\|}{|\ell_{ij} \widehat{u}_{jj}|} \end{cases}$$

Then $\|A - \widehat{L}\widehat{U}\| \leq \varepsilon \|A\|$

AP incomplete factorization (ongoing work w/ Lister and Scott)

- Let $A = LU$; ℓ_{ij} is computed as

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} u_{kj}\right) / u_{jj}$$

- **Analysis:** let $\widehat{\ell}_{ij} = \ell_{ij}(1+\delta_{ij})$; we have

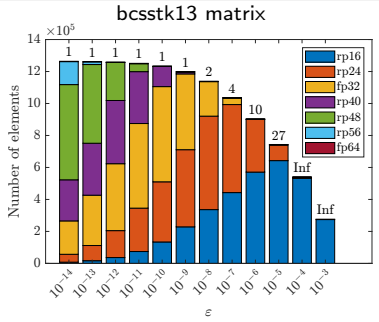
$$\ell_{ij}(1+\delta_{ij}) = \left(a_{ij} - \sum_{k=1}^{j-1} \widehat{\ell}_{ik}\widehat{u}_{kj}\right) / \widehat{u}_{jj}$$

$$\Leftrightarrow a_{ij} = \sum_{k=1}^{j} \widehat{\ell}_{ik}\widehat{u}_{kj} + \ell_{ij}\widehat{u}_{jj}\delta_{ij}$$

$\Rightarrow$ **Algorithm:**

$$\begin{cases} \text{if } |\ell_{ij}\widehat{u}_{jj}| \le \varepsilon\|A\| & \text{drop } \ell_{ij} \ (\widehat{\ell}_{ij} = 0,\ \delta_{ij} = -1) \\ \text{else} & \text{store } \ell_{ij} \text{ in precision } \le \frac{\varepsilon\|A\|}{|\ell_{ij}\widehat{u}_{jj}|} \end{cases}$$

Then $\|A - \widehat{L}\widehat{U}\| \le \varepsilon\|A\|$



AP incomplete factorization (ongoing work w/ Lister and Scott)

- Let $A = LU$; ℓ_{ij} is computed as

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} u_{kj}\right) / u_{jj}$$

- **Analysis:** let $\widehat{\ell}_{ij} = \ell_{ij}(1+\delta_{ij})$; we have

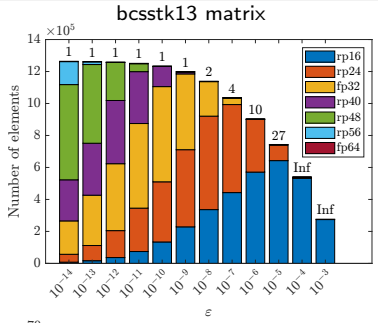
$$\ell_{ij}(1+\delta_{ij}) = \left(a_{ij} - \sum_{k=1}^{j-1} \widehat{\ell}_{ik}\widehat{u}_{kj}\right) / \widehat{u}_{jj}$$

$$\Leftrightarrow a_{ij} = \sum_{k=1}^{j} \widehat{\ell}_{ik}\widehat{u}_{kj} + \ell_{ij}\widehat{u}_{jj}\delta_{ij}$$

$\Rightarrow$ **Algorithm:**

$$\begin{cases} \text{if } |\ell_{ij}\widehat{u}_{jj}| \le \varepsilon\|A\| & \text{drop } \ell_{ij} \ (\widehat{\ell}_{ij} = 0,\ \delta_{ij} = -1) \\ \text{else} & \text{store } \ell_{ij} \text{ in precision } \le \frac{\varepsilon\|A\|}{|\ell_{ij}\widehat{u}_{jj}|} \end{cases}$$

Then $\|A - \widehat{L}\widehat{U}\| \le \varepsilon\|A\|$



AP incomplete factorization (ongoing work w/ Lister and Scott)

- Let $A = LU$; ℓ_{ij} is computed as

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} u_{kj}\right) / u_{jj}$$

- **Analysis:** let $\widehat{\ell}_{ij} = \ell_{ij}(1+\delta_{ij})$; we have

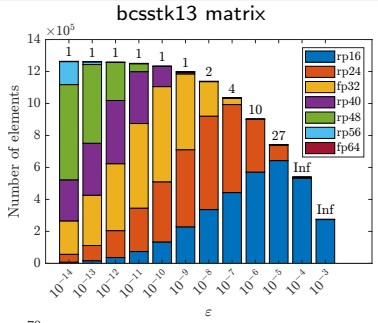
$$\ell_{ij}(1+\delta_{ij}) = \left(a_{ij} - \sum_{k=1}^{j-1} \widehat{\ell}_{ik}\widehat{u}_{kj}\right) / \widehat{u}_{jj}$$

$$\Leftrightarrow a_{ij} = \sum_{k=1}^{j} \widehat{\ell}_{ik}\widehat{u}_{kj} + \ell_{ij}\widehat{u}_{jj}\delta_{ij}$$

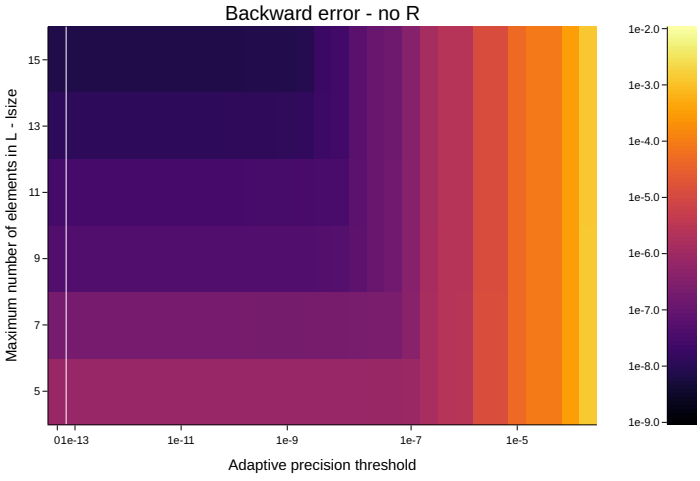
$\Rightarrow$ **Algorithm:**

$$\begin{cases} \text{if } |\ell_{ij}\widehat{u}_{jj}| \le \varepsilon\|A\| & \text{drop } \ell_{ij} \ (\widehat{\ell}_{ij} = 0,\ \delta_{ij} = -1) \\ \text{else} & \text{store } \ell_{ij} \text{ in precision } \le \frac{\varepsilon\|A\|}{|\ell_{ij}\widehat{u}_{jj}|} \end{cases}$$

Then $\|A - \widehat{L}\widehat{U}\| \le \varepsilon\|A\|$



AP incomplete factorization (ongoing work w/ Lister and Scott)

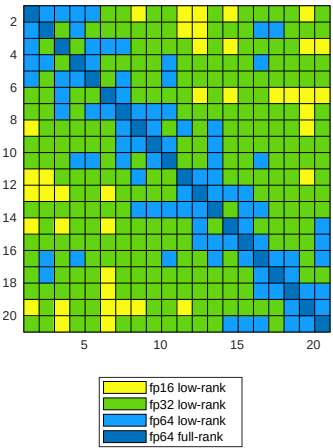


crystm01 matrix

- HSL_MI28: memory-limited factorization with fixed number of elements per column
- Preconditioner error ε given by largest element dropped $\Rightarrow$ use same ε for adaptive precision to balance errors!

AP tile factorization (ongoing work w/ Abdurakhmanov, Ltaief, Keyes, Fahmy)

Root frontal matrix
in 3D Poisson problem

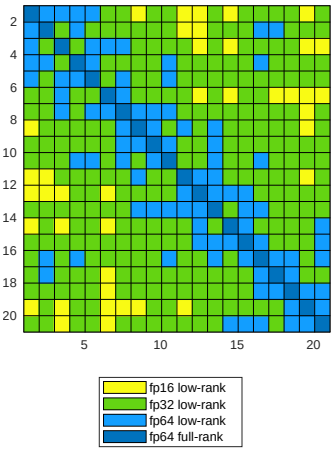


- Consider a $pb \times pb$ matrix A partitioned into $b \times b$ tiles A_{ij}
- **Analysis:** let $\widehat{A}$ be obtained by rounding each A_{ij} to precision u_{ij}. Then we have

$$\|A - \widehat{A}\|_F^2 = \sum_{i,j} \|A_{ij} - \widehat{A}_{ij}\|_F^2$$
$$\leq \sum_{i,j} u_{ij}^2 \|A_{ij}\|_F^2$$

AP tile factorization (ongoing work w/ Abdurakhmanov, Ltaief, Keyes, Fahmy)

Root frontal matrix
in 3D Poisson problem



- Consider a $pb \times pb$ matrix A partitioned into $b \times b$ tiles A_{ij}
- **Analysis:** let $\widehat{A}$ be obtained by rounding each A_{ij} to precision u_{ij}. Then we have

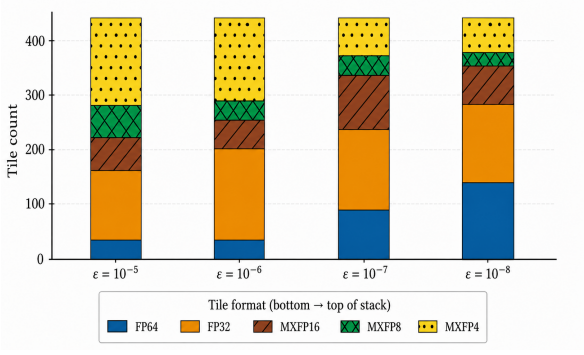
$$\|A - \widehat{A}\|_F^2 = \sum_{i,j} \|A_{ij} - \widehat{A}_{ij}\|_F^2 \leq \sum_{i,j} u_{ij}^2 \|A_{ij}\|_F^2$$

$\Rightarrow$ **Algorithm:** store A_{ij} in precision $u_{ij} \propto \frac{\|A\|}{\|A_{ij}\|}\varepsilon$

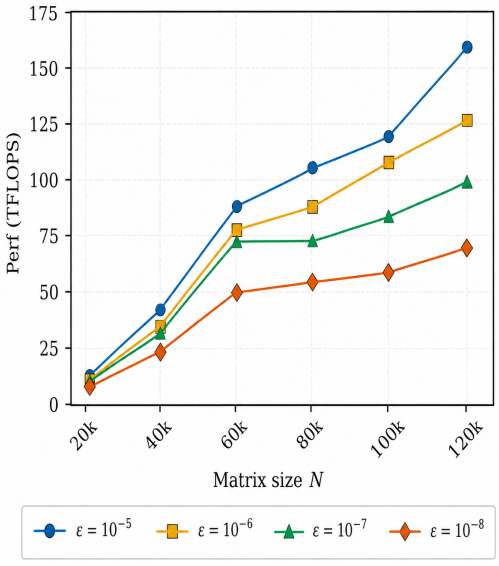
- Moreover, in the factorization of A, the update $A_{ij} \leftarrow A_{ij} - A_{ik}A_{kj}$ can be performed in precision $u_{ijk} \propto \frac{\|A\|}{\|A_{ik}\|\|A_{kj}\|}\varepsilon$

AP tile factorization (ongoing work w/ Abdurakhmanov, Ltaief, Keyes, Fahmy)

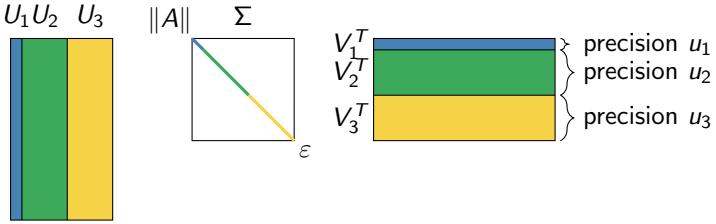
Results on weak-correlation Matérn covariance matrices on a single NVIDIA GB200 GPU



MX formats use microscaling to prevent underflow



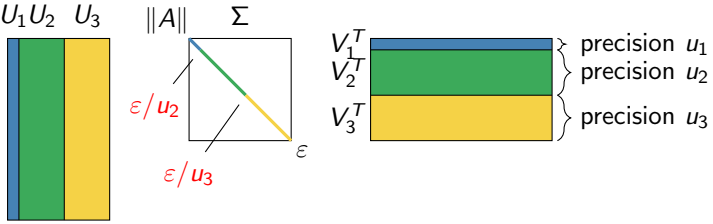
AP low-rank approximation (Amestoy, Boiteau, Buttari, Gerest, Jézéquel, L'Excellent, M., 2022)



- **Analysis**: Given $A = U\Sigma V^T$, define the adaptive precision partitioning $\widehat{A} = \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T$ where $\widehat{U}_k$ and $\widehat{V}_k$ are stored in precision u_k. Then

$$\|A - \widehat{A}\| \leq \sum_{k=1}^{p} c u_k \|\Sigma_k\|$$

AP low-rank approximation (Amestoy, Boiteau, Buttari, Gerest, Jézéquel, L'Excellent, M., 2022)



- **Analysis**: Given $A = U\Sigma V^T$, define the adaptive precision partitioning $\widehat{A} = \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T$ where $\widehat{U}_k$ and $\widehat{V}_k$ are stored in precision u_k. Then

$$\|A - \widehat{A}\| \leq \sum_{k=1}^{p} c u_k \|\Sigma_k\|$$

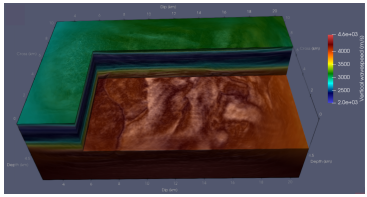
$\Rightarrow$ **Algorithm**: sort Σ and partition it such that $\|\Sigma_k\| \leq \varepsilon \|A\|/u_k$, yielding $\|\widehat{A} - A\| \leq c'\varepsilon\|A\|$

- Beyond SVD, can be applied to any LRA with decaying rank-1 components (e.g., RRQR)

AP block low-rank approximation

- We use AP BLR approximations in MUMPS for reducing the LU factor storage cost
- We use 7 custom RP precisions with truncated mantissa and a memory accessor to perform the LU solves (Amestoy, Jego, L'Excellent, M., Pichon, 2025)
- Illustrative application impact (Operto et al., 2023):

Gorgon Model, reservoir 23km x 11km x 6.5km,
grid size 15m, Helmholtz equation, 25-Hz
Complex matrix, 531 Million dofs



FR (Full-Rank); BLR with $\varepsilon = 10^{-5}$; 48 000 cores (500 MPI $\times$ 96 threads/MPI)
FR: fp32; AP BLR: 3 precisions (fp32, rp24, rp16) for storage

LU size (TBytes)			Flops		Time BLR + Mixed (sec)			Scaled Resid.
FR	BLR	+adapt.	FR	BLR+adapt.	Analysis	Facto	Solve	BLR+adapt.
73	34	26	2.6×10^{18}	0.5×10^{18}	446	5500	27	7×10^{-4}